Computational assessment of curvatures and principal directions of implicit surfaces from 3D scalar data

Eric Albin, Ronnie Knikker, Shihe Xin, Christian Oliver Paschereit, Yves d'Angelo

Computational assessment of curvatures and principal directions of implicit surfaces from 3D scalar data

Eric Albin[1], Ronnie Knikker[1], Shihe Xin[1], Christian Oliver Paschereit[2], and Yves D'Angelo[3]

[1] Université de Lyon, CNRS
Université Lyon 1, F-69622, France
INSA-Lyon, CETHIL, UMR5008, F-69621, Villeurbanne, France
eric.albin@univ-lyon1.fr,
[2] Technische Universität Berlin, Institute of Fluid Dynamics and Technical Acoustics, Hermann-Föttinger-Institut. Müller-Breslau-Str. 8 10623 Berlin, Germany
[3] Laboratoire de Mathématiques J.A. Dieudonné, CNRS UMR 7351, Université de Nice Sophia-Antipolis, Parc Valrose 06108 Nice, France

Abstract. An implicit method based on high-order differentiation to determine the mean, Gaussian and principal curvatures of implicit surfaces from a three-dimensional scalar field is presented and assessed. The method also determines normal vectors and principal directions. Compared to explicit methods, the implicit approach shows robustness and improved accuracy to measure curvatures of implicit surfaces. This is evaluated on simple cases where curvature is known in closed-form. The method is applied to compute the curvatures of wrinkled flames on large triangular unstructured meshes (namely a 3D isosurface of temperature).

Keywords: implicit surface, curvature, principal directions, isosurface, 3D scalar field, combustion analysis

Introduction

Curvatures play an important role in many areas of physics where interfaces are encountered [13,20]. For instance, local curvatures can modify combustion speed [1,8,9,38], surface tension [4] and evaporation speed [7]. Accurate methods for determining curvatures can then be of paramount importance for flow analysis. In the recent study of Yu *et* Bai [38], local mean and Gaussian curvatures are computed to analyse deflagration fronts during an auto-ignition. Strong saddle front and small sphere front are shown to play an important role during the auto-ignition process. However, numerical techniques used to estimate curvatures are not detailed and plots with curvatures show a large scattering of data. The present work proposes implicit methods to improve such analyses. These methods measure curvatures and principal directions of isosurfaces extracted from 3D data. Many methods use an explicit representation of surfaces, where

the surface is discretized by elementary 2D cells like triangles, but very few make use of values of the scalar function that defines implicitly the surface.

The visualization of implicit surfaces is however commonly encountered in magnetic resonance, tomography [19] or in the post-processing of three dimensional data of numerical simulation [31]. The implicit description of surfaces is also used in level set methods [26] where a scalar function $\phi(x, y, z)$ is introduced for computational purpose. The use of such an implicit representation can handle complex changes of topology that may be encountered e.g. in primary break-up of sprays [17] or in turbulent flame propagation [36]. The dynamics of explicit three-dimensional flame surfaces can be computed, but requires complex treatment of intersections (coalescence, break-up) [9]. Whereas the mean curvature is often deduced from the divergence of $\boldsymbol{\nabla}\phi$ [26], the Gaussian curvature is usually not computed. However, a similar expression for the Gaussian curvature for implicitly defined surfaces, provided in the recent studies of Trott [35] and Goldman [14], is of interest to deduce minimal and maximal curvatures for flow analysis. Note that Goldman also extends curvature formulas to higher dimension than 3D but does not discuss about principal directions. In [18], Lehmann *et* Reif detail how to build the curvature tensor. No explicit formulas are given for principal curvatures or principal directions but their methodology allows to compute them independently of a given parametrization by searching eigenvalues and eigenvectors. In our study, a scalar formulation of Goldmann's expression and an algorithm derived from [18] are validated through numerical tests with high order diffuse approximation and Lagrange differentiation schemes.

Implicit surfaces of 3D scalar data are commonly extracted by a marching cube method to be visualized or analysed. This method has been introduced by Lorensen and Cline [19] to extract an explicit unstructured mesh of the isosurface using triangles including topological information. The obtained isocontour is a three-dimensional surface. It is for instance used for flow visualization in the open-source software ParaView [31] or in the GTS library [29,34]. Various algorithms are available to determine curvatures from a triangle mesh [10,15,22,24,27,28]. Such algorithms may be used to compute curvatures of the explicit mesh representing the isosurface. We call them by "*explicit methods*" in opposition to "*implicit methods*" that use 3D scalar data. A comparison between many usual methods can be found in Gatze *et* al. [12]. For instance, local fitting methods of analytical surface with a least square approximation allow to approximate mean and Gaussian curvatures [12,22]. Meyer *et* al. [24] define several discrete operators to compute normal vectors and curvatures. If discrete methods [11,24] are appealing from a computational point of view compared to fitting methods, they suffer from the noise and regularity issues [12]. In this paper, we show that the present 'implicit method' to determine curvatures is more accurate and less sensitive to surface discretization errors than these 'explicit methods' to analyse isosurfaces from 3D scalar data.

In section 1, we gather some formula of the literature allowing to compute the principal curvatures of implicit surfaces and propose an algorithm to determine corresponding principal directions. High order differentiation schemes

used in 'implicit methods' are briefly presented in section 2. The accuracy of the proposed implicit methods to determine curvatures, normals and principal directions is assessed in section 3. The method is also tested to measure curvatures of isocontours extracted from 3D scalar fields and compared to 'explicit methods' in terms of accuracy and speed. It is then applied on large data of Direct Numerical Simulations of expanding and imploding flames in section 4. Concluding remarks end the paper in section 5.

1 Curvatures and principal directions of implicit surfaces

In level set methods, it is common to use a 3D scalar function $\phi(x, y, z)$ to implicitly define an interface. The interface is then a 3D surface defined as the isocontour $\phi(x, y, z) = \phi_0 = \text{cst}$. Some geometrical properties like normals $\boldsymbol{n}$ and mean curvatures κ_H are deduced from ϕ using formulas (1) and (2):

$$\boldsymbol{n} = \frac{\boldsymbol{\nabla}\phi}{\|\boldsymbol{\nabla}\phi\|} = \frac{\boldsymbol{\nabla}\phi}{[(\partial\phi/\partial x)^2 + (\partial\phi/\partial y)^2 + (\partial\phi/\partial z)^2]^{1/2}} \tag{1}$$

$$\begin{aligned}\kappa_H = \frac{\boldsymbol{\nabla}\cdot\boldsymbol{n}}{2} &= \frac{\phi_x^2(\phi_{yy}+\phi_{zz}) + \phi_y^2(\phi_{xx}+\phi_{zz}) + \phi_z^2(\phi_{xx}+\phi_{yy})}{2\cdot(\phi_x^2+\phi_y^2+\phi_z^2)^{3/2}} \\ &\quad - \frac{\phi_x\phi_y\phi_{xy} + \phi_x\phi_z\phi_{xz} + \phi_y\phi_z\phi_{yz}}{(\phi_x^2+\phi_y^2+\phi_z^2)^{3/2}} \qquad (2a) \\ &= \frac{\phi_{uu}+\phi_{vv}}{2\cdot|\phi_n|} \qquad (2b)\end{aligned}$$

The two-dimensional version of equation (2a) was first used in the context of the level set method by Chang *et* al. [6]. The three-dimensional version can be found for example in Osher and Fedkiw [26, pp. 12]. In an orthonormal frame ($\boldsymbol{n}$, $\boldsymbol{u}$, $\boldsymbol{v}$) where $\boldsymbol{u}$ and $\boldsymbol{v}$ are arbitrary vectors defining the tangent plane to the surface, the formula (2a) reduces to a simpler expression (2b).

A similar expression for the Gaussian curvature κ_K is derived in Goldman (see eqn (4.1) in [14]) where κ_K is expressed as a function of the gradient $\boldsymbol{\nabla}\phi$, the Hessian matrix $H(\phi)$ and its adjugate matrix. A scalar formulation is also derived using the Mathematica software in Trott [35, pp. 1285–1286]. The formulation (3a) used in this work is a scalar formulation similar to [35] but slightly reformulated to prevent a division by zero when ϕ_z vanishes:

$$\begin{aligned}\kappa_K = 2&\frac{\phi_x\phi_y(\phi_{xz}\phi_{yz} - \phi_{xy}\phi_{zz}) + \phi_x\phi_z(\phi_{xy}\phi_{yz} - \phi_{xz}\phi_{yy}) + \phi_y\phi_z(\phi_{xy}\phi_{xz} - \phi_{yz}\phi_{xx})}{(\phi_x^2+\phi_y^2+\phi_z^2)^2} \\ &+ \frac{\phi_x^2\cdot(\phi_{yy}\phi_{zz} - \phi_{yz}^2) + \phi_y^2\cdot(\phi_{xx}\phi_{zz} - \phi_{xz}^2) + \phi_z^2\cdot(\phi_{xx}\phi_{yy} - \phi_{xy}^2)}{(\phi_x^2+\phi_y^2+\phi_z^2)^2} \qquad (3a) \\ &= \frac{\phi_{uu}\phi_{vv} - \phi_{uv}^2}{\phi_n^2} \qquad (3b)\end{aligned}$$

Expressions (2b) and (3b) are also the eigenvalues of the curvature tensor as explained in [18] and shown in the annex A. The annex A also shows how to derive expressions in the $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})$ basis from ϕ_{uu}, ϕ_{uv} and ϕ_{vv} using change of basis (15). Since $\kappa_K = \kappa_{min} \cdot \kappa_{max}$ and $\kappa_H = (\kappa_{min} + \kappa_{max})/2$, the minimal and maximal curvatures are then deduced from the mean and Gaussian curvatures:

$$\kappa_{min} = \kappa_H - \sqrt{|\kappa_H^2 - \kappa_K|} \tag{4a}$$

$$\kappa_{max} = \kappa_H + \sqrt{|\kappa_H^2 - \kappa_K|} \tag{4b}$$

With the chosen convention of orientation, normal vectors point to large values of ϕ and curvatures are negative when normals converge towards high ϕ values.

At umbilical points where $\kappa_{min} = \kappa_{max}$, any direction is a principal direction. However, directions of the minimal and maximal curvatures of implicit surfaces are single solutions at non-umbilical points and may be computed by searching eigenvectors of the curvature tensor, as detailed in Lehmann *et* Reif [18]. Rather than using an eigenvector solver, we derive in annex A expressions to directly compute principal directions $\boldsymbol{t_1}$, $\boldsymbol{t_2}$ corresponding to principal curvatures $\kappa_1 = \kappa_H - \sqrt{|\kappa_H^2 - \kappa_K|} \cdot \zeta$, $\kappa_2 = \kappa_H + \sqrt{|\kappa_H^2 - \kappa_K|} \cdot \zeta$:

$$\boldsymbol{t_1} = \begin{bmatrix} 0 \\ \phi_{uv} \\ \kappa_1\phi_n - \phi_{uu} \end{bmatrix}_{\boldsymbol{n},\boldsymbol{u},\boldsymbol{v}} = \begin{bmatrix} (\kappa_1\phi_n - \phi_{uu}) \cdot v_x + \phi_{uv} u_x \\ (\kappa_1\phi_n - \phi_{uu}) \cdot v_y + \phi_{uv} u_y \\ (\kappa_1\phi_n - \phi_{uu}) \cdot v_z + \phi_{uv} u_z \end{bmatrix}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}} \tag{5a}$$

$$\boldsymbol{t_2} = \begin{bmatrix} 0 \\ \kappa_2\phi_n - \phi_{vv} \\ \phi_{uv} \end{bmatrix}_{\boldsymbol{n},\boldsymbol{u},\boldsymbol{v}} = \begin{bmatrix} (\kappa_2\phi_n - \phi_{vv}) \cdot u_x + \phi_{uv} v_x \\ (\kappa_2\phi_n - \phi_{vv}) \cdot u_y + \phi_{uv} v_y \\ (\kappa_2\phi_n - \phi_{vv}) \cdot u_z + \phi_{uv} v_z \end{bmatrix}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}} \tag{5b}$$

Principal direction coordinates into brackets are given both in the $(\boldsymbol{n},\boldsymbol{u},\boldsymbol{v})$ and $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})$ bases. The subscript notations in (5a) indicate the coordinate system used for the vector decomposition. These expressions are quite similar to those found in the litterature dealing with the computation of crest lines [5,25,33], except we introduce a sign function $\zeta = \pm 1$ to circumvent degeneracies (see case studies proposed at the end of annex A). Because a wrong choice of $\boldsymbol{u}$, $\boldsymbol{v}$ may yield to find $\boldsymbol{t_1} = \boldsymbol{t_2} = \boldsymbol{0}$, we propose a criterion on ζ to ensure that the found vectors are non-null for any arbitrary choice of $\boldsymbol{u}$, $\boldsymbol{v}$. Since equation (2b) implies $|\kappa_{min}\phi_n - \phi_{uu}| = |\kappa_{max}\phi_n - \phi_{vv}|$, we propose the following algorithm in order to compute $\boldsymbol{t_{min}}$ and $\boldsymbol{t_{max}}$ associated to κ_{min} and κ_{max}:

choose arbitrarily $\boldsymbol{u}$, $\boldsymbol{v}$ from $\boldsymbol{n}$;
compute ϕ_{uu} and ϕ_{vv} with system (15);
if $|\kappa_{min}\phi_n - \phi_{uu}| \geqslant |\kappa_{min}\phi_n - \phi_{vv}|$ **then**
  choose $\zeta = +1$ to avoid $\kappa_1\phi_n - \phi_{uu} = -(\kappa_2\phi_n - \phi_{vv}) = 0$;
  compute $\boldsymbol{t_{min}} = \boldsymbol{t_1}$ and $\boldsymbol{t_{max}} = -\boldsymbol{t_2}$ with $\kappa_{min} = \kappa_1$ and $\kappa_{max} = \kappa_2$;
else
  choose $\zeta = -1$ since $|\kappa_{max}\phi_n - \phi_{uu}| > |\kappa_{max}\phi_n - \phi_{vv}| \geqslant 0$;
  compute $\boldsymbol{t_{min}} = -\boldsymbol{t_2}$ and $\boldsymbol{t_{max}} = \boldsymbol{t_1}$ with $\kappa_{max} = \kappa_1$ and $\kappa_{min} = \kappa_2$;
end

This algorithm requires to arbitrarily choose ($\boldsymbol{u}$,$\boldsymbol{v}$) and compute ϕ_{uu} and ϕ_{vv} to determine κ_H, κ_K, $\boldsymbol{t_1}$ and $\boldsymbol{t_2}$ with explicit expressions (2b), (3b), (5a) and (5b). An alternative choice would be to build the curvature tensor in standard coordinates [18] and use a specific solver for searching eigenvalues and eigenvectors. If principal directions are not required, curvatures are computed by the *intrinsic* formulas (2a) and (3a) that do not require any choice of $\boldsymbol{u}$, $\boldsymbol{v}$ or eigen solver. Some exercises are also proposed at the end of annex A to better understand the following algorithm. This algorithm implies that the basis ($\boldsymbol{t_{min}}$,$\boldsymbol{t_{max}}$,$\boldsymbol{n}$) is direct and:

$$\kappa_{min} = \kappa_1 \tfrac{1+\zeta}{2} + \kappa_2 \tfrac{1-\zeta}{2} \qquad \kappa_{max} = \kappa_1 \tfrac{1-\zeta}{2} + \kappa_2 \tfrac{1+\zeta}{2} \tag{6a}$$

$$\boldsymbol{t_{min}} = \boldsymbol{t_1} \tfrac{1+\zeta}{2} + \boldsymbol{t_2} \tfrac{1-\zeta}{2} \qquad \boldsymbol{t_{max}} = \boldsymbol{t_1} \tfrac{1-\zeta}{2} - \boldsymbol{t_2} \tfrac{1+\zeta}{2} \tag{6b}$$

2 High order differentiation schemes for implicit methods

In order to compute curvatures and principal directions using equations (1) to (5), it is necessary to compute second-order cross derivatives of ϕ at a given point $P(x, y, z)$ from neighbor points P_i where ϕ is discretized. The point P where curvatures have to be computed is not necessarily lying on one of the nodes P_i. Figure 1 shows a schematic discretization of ϕ.

Fig. 1a non Cartesian stencil

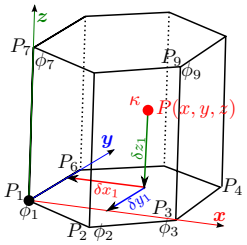

Fig. 1b Cartesian stencil

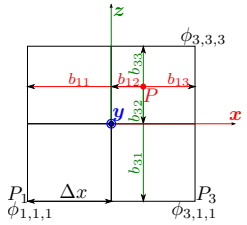

Fig. 1. Schematic representation of a point P where curvatures are computed from a 3D discrete scalar field ϕ_j. Coefficients (δx_j, δy_j, δz_j) or b_{ij} represent the position of the neighbor discrete points P_j to P.

2.1 Diffuse approximation (DA)

Diffuse approximations (DA) allow to compute all derivatives of ϕ at P from the values ϕ_i known at points P_i of coordinates $(x+\delta x_i, y+\delta y_i, z+\delta z_i)$ with any kind of mesh [30]. Coefficients δx_i, δy_i and δz_i are then small real numbers defining the position of the i^{th} neighbor point P_i relatively to P (see Fig. 1a). The value of the scalar at a point P_i is estimated by a Taylor expansion $\phi_i^* = \sum_{j=1}^{N_u} P_{ij}\alpha_j$. The

matrices $\boldsymbol{P}$ and $\boldsymbol{\alpha}$ are given as an example for a second-order Taylor expansion ($N = 2$, $N_u = 10$) but can be similarly constructed for a fourth order $N = 4$:

$$\boldsymbol{P} = [P_{ij}] = \left[1,\ \delta x_i,\ \delta y_i,\ \delta z_i,\ \delta x_i^2,\ \delta y_i^2,\ \delta z_i^2,\ \delta x_i \delta y_i,\ \delta x_i \delta z_i,\ \delta y_i \delta z_i\right] \quad (7a)$$

$$\boldsymbol{\alpha} = [\alpha_j] \ = \left[\phi\ ,\ \phi_x\ ,\ \phi_y\ ,\ \phi_z\ ,\ \frac{\phi_{xx}}{2}\ ,\ \frac{\phi_{yy}}{2}\ ,\ \frac{\phi_{zz}}{2}\ ,\ \phi_{xy}\ ,\ \phi_{xz}\ ,\phi_{yz}\right] \quad (7b)$$

If a number N_n of neighbor nodes are used around P, the method consists in minimizing the quadratic error $I(\boldsymbol{\alpha}) = \sum_{i=1}^{N_n} \omega_i (\phi_i - \phi_i^*)^2$ weighted by $\omega_i = e^{-(\delta x_i^2 + \delta y_i^2 + \delta z_i^2)/\Delta x^2}$. This minimization yields a linear system $\boldsymbol{A} \cdot \boldsymbol{\alpha} = \boldsymbol{B}$ to solve with:

$$\boldsymbol{A} = [A_{kj}] = \left[\sum_{i=1}^{N_n} \omega_i \cdot P_{ik} \cdot P_{ij}\right] \qquad \text{with} \qquad j \in [1; N_u] \quad (8a)$$

$$\boldsymbol{B} = [B_k] = \left[\sum_{i=1}^{N_n} \omega_i \cdot P_{ik} \cdot \phi_i\right] \quad (8b)$$

$\boldsymbol{P}$ is a matrix of size $N_n \times N_u$. The searched vector $\boldsymbol{\alpha}$ of size N_u is then computed after inverting the matrix $\boldsymbol{A}$ of size N_u^2. Minimizing the quadratic error with a Taylor expansion of order $N + 1$ implies a N^{th}-order accuracy for the first derivatives and an order $N - 1$ for the second derivatives.

Note that Marchandise et al [21], in the context of two-phase flows, already used a least-squares method (2^{nd}-order DA with no weighting function : $\omega_i = 1$) and equation $\kappa_H = -\boldsymbol{\nabla} \cdot \boldsymbol{n}$ to compute the mean curvature with a slightly different two-step procedure. A 2^{nd}-order DA method is also used in the coupled VOF-level set method proposed by Sussmann *et* al. [23,32], but only to compute the normal of the piecewise linear surface reconstructions. In this paper, the DA method is used directly to compute principal curvatures and directions with higher order accuracy at any point on the interface.

2.2 Lagrange differentiation (LD)

High-order differentiation with Lagrange polynomials are also tested to compute principal curvatures and directions if the implicit function ϕ is discretized on a Cartesian grid (Fig. 1b shows a Cartesian stencil as an example with $n = 3$). First, second and cross derivatives like ϕ_x, ϕ_{zz} and ϕ_{xy} are computed by weighting neighbour values of the implicit function over the Cartesian stencil:

$$\left.\frac{\partial \phi}{\partial x_1}\right|_P = \phi_x = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \dot{\beta}_{1i} \cdot \bar{\beta}_{2j} \cdot \bar{\beta}_{3k} \cdot \phi_{i,j,k} \quad (9a)$$

$$\left.\frac{\partial \phi}{\partial x_3^2}\right|_P = \phi_{zz} = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \bar{\beta}_{1i} \cdot \bar{\beta}_{2j} \cdot \ddot{\beta}_{3k} \cdot \phi_{i,j,k} \quad (9b)$$

$$\left.\frac{\partial^2 \phi}{\partial x_1 \partial x_2}\right|_P = \phi_{xy} = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \dot{\beta}_{1i} \cdot \dot{\beta}_{2j} \cdot \bar{\beta}_{3k} \cdot \phi_{i,j,k} \quad (9c)$$

Weighting coefficients $\bar{\beta}_{ij}$, $\dot{\beta}_{ij}$ or $\ddot{\beta}_{ij}$ correspond respectively to an interpolation, a first differentiation or a second differentiation in the $\boldsymbol{x_i}$ direction. Their expression are similar to Lagrange basis polynomials [16] and their derivatives, they are obtained from the inversion of the Vandermonde matrix [2]:

$$\bar{\beta}_{ij} = \Big(\prod_{\substack{k=1 \\ k\neq j}}^{n} b_{ik} \Big) / A_{ij} \qquad \text{with} \qquad A_{ij} = \prod_{\substack{k=1 \\ k\neq j}}^{n} (b_{ik} - b_{ij}) \tag{10a}$$

$$\dot{\beta}_{ij} = -\Big(\sum_{\substack{k=1 \\ k\neq j}}^{n} \prod_{\substack{l=1 \\ l\neq j,k}}^{n} b_{il} \Big) / (A_{ij} \cdot \Delta x) \tag{10b}$$

$$\ddot{\beta}_{ij} = \Big(\sum_{\substack{k=1 \\ k\neq j}}^{n} \sum_{\substack{l=1 \\ l\neq j,k}}^{n} \prod_{\substack{m=1 \\ m\neq j,k,l}}^{n} b_{im} \Big) / (A_{ij} \cdot \Delta x^2) \tag{10c}$$

These coefficients can be directly computed from b_{ij} that represents the position of the point P in the $\boldsymbol{x_i}$ direction within the stencil (see Fig. 1b). As an example, the reader may compute $\bar{\beta}_{ij}$, $\dot{\beta}_{ij}$ or $\ddot{\beta}_{ij}$ for $j \in [1, 2, n = 3]$ with $b_{i1} = -\Delta x$, $b_{i2} = 0$, $b_{i3} = \Delta x$ to recover well-known coefficients of finite differences.

3 Test of implicit methods

3.1 Accuracy assessment of implicit methods

A spherical surface with known curvature is used to test the accuracy of the method. The radius of this sphere centered at the point C is denoted R and the parameter δ controls the stiffness of the scalar function using a hyperbolic tangent profile:

$$\begin{aligned} \phi(x,y,z) &= \tanh\left((r-R)/\delta\right) \\ r &= \sqrt{(x-x_C)^2 + (y-y_C)^2 + (z-z_C)^2} \\ x_C &= x_O + R\sin\phi\cos\theta \\ y_C &= y_O + R\sin\phi\sin\theta \\ z_C &= z_O + R\cos\phi \end{aligned} \tag{11}$$

Figure 2 shows a schematic representation of this spherical function around a cubic stencil of center O. In the present analysis, the stencils are defined on Cartesian grids, with $N_n = n^3$ points in order to allow comparisons with LD and DA differentiation schemes. The implicit spherical function is characterized by its radius R, angles (θ, ϕ) and stiffness $1/\delta$ in equation (11).

The accuracy of the implicit method for computing curvatures is investigated by varying the surface normal via (θ, ϕ) and the spherical function via R and δ. Figure 3a shows the relative error on the mean-curvature $\Delta\kappa_H/\kappa_H = \kappa_H \cdot R - 1$ determined at point O when varying the normal angles ($\theta \in [0; 2\pi]$, $\phi \in [0; \pi]$,

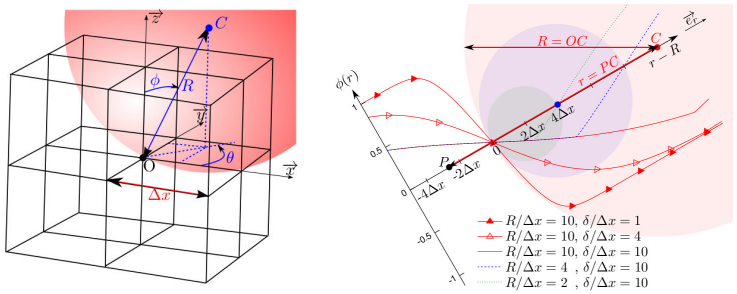

Fig. 2. Representation of a volumetric spherical function with known curvatures.

$R/\Delta x = 10$, $\delta/\Delta x = 2$) with the second-order diffuse approximation method ($N = 2$, $N_n = 5^3$). Because the expressions (2)-(3) and the stencil present a symmetry towards planes (O, $\overrightarrow{x}$), (O, $\overrightarrow{y}$) and (O, $\overrightarrow{z}$), the error is periodical along θ and ϕ directions and the study of errors could be limited to one eighth of the domain. The maximal error is obtained at some positions of the curvature center and can be extracted from these curves; it will be denoted $\max|\Delta\kappa_H/\kappa_H|$. The maximal error on this domain is subsequently plotted for various radius R and stiffness $1/\delta$ in Fig. 3b. The error is higher for stiff or highly-curved functions. The stiffness and the curvature have a similar influence on the error that is then controlled by the most critical parameter between high curvatures and high stiffness.

Fig. 3a

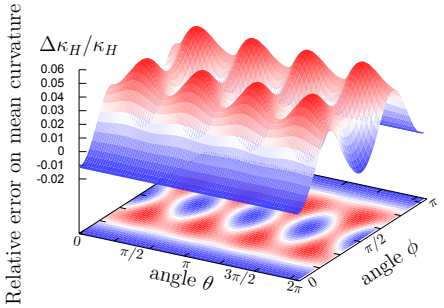

Fig. 3b

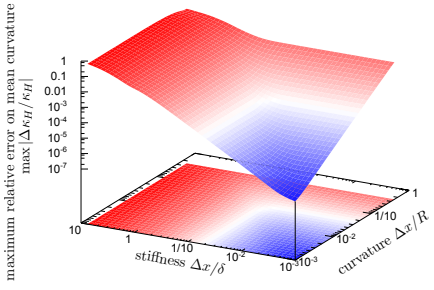

Fig. 3. Measurements of the relative error when computing the mean curvature at the center O of a stencil of size $\pm 2\Delta x$ with a second-order diffuse approximation.

Figure 4 shows the maximum relative errors on the mean curvatures computed at the center of the stencil as a function of the critical parameters (stiffness or curvature) to compare the effect of the numerical schemes on the error. The obtained error on Gaussian curvatures are really similar to mean curvature errors and are not shown for brevity. Errors decrease for smooth function and weakly

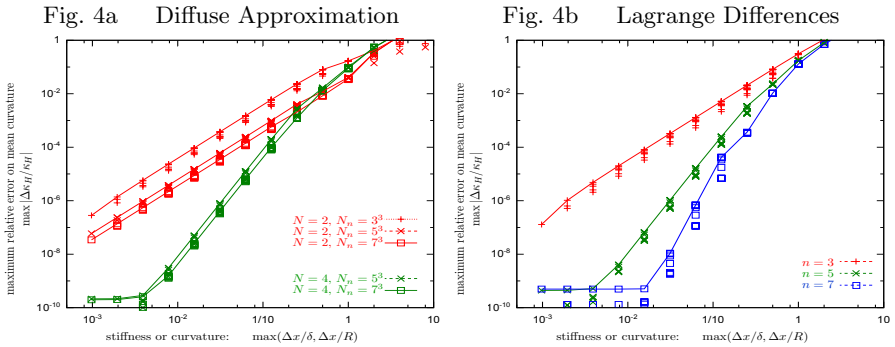

Fig. 4. Relative error on mean curvatures for various stencil sizes ($N_n = n^3$ with $n \in \{3; 5; 7\}$) and numerical schemes (accuracy order: N for DA and n for LD schemes).

curved surfaces. Implicit methods with Lagrange differentiation (LD) and diffuse approximation (DA) present similar accuracies. For a second-order DA ($N = 2$) with $N_n = 3^3$ points, the error is similar to the 2^{nd}-order LD ($n = 3$). However, this error decreases when the stencil is larger with second-order DA ($N_n = 5^3$ or $N_n = 7^3$). Accuracies on curvatures increase for high-order LD or DA numerical schemes. They are similar with 4^{th} order schemes (DA with $N = 4$, LD with $n = 5$). The implicit method with a 6^{th}-order LD scheme for first-derivatives ($n = 7$) reaches lowest errors in this test where no interpolation inside stencils is performed.

3.2 Accuracy comparison using a marching cube extraction of isocontours

The accuracy of proposed implicit methods is now compared to standard explicit methods when computing mean and Gaussian curvatures of a sphere with varying sphere resolution $R/\Delta x$. In this test, the initial data is for all cases an implicit 3D spherical function defined by equation (11) and centered at $x_C = y_C = z_C = 0$ in a cubic box of size $[-\frac{N}{2}\Delta x; \frac{N}{2}\Delta x]$ large enough to contain the radius R and stencil points; the stiffness is chosen equal to $\frac{\Delta x}{R}$. As shown in Fig. 5a, an explicit irregular triangulated surface of this sphere of radius R is extracted using a marching cube method [19]. Fig. 5b illustrates how triangle meshes are generated from linear interpolation on edges of the Cartesian grid with a marching cube method (cf. edges P_1P_2, P_1P_3, P_5P_6 and P_5P_7).

Mean and Gaussian curvatures are then computed at the nodes of this spherical isocontour with various implicit and explicit methods for comparisons. The first explicit method is classical and consists in fitting full quadric [12,22]. Discrete explicit methods implemented in the ParaView software [31] or the GTS library [29] are also tested. The mean curvature is computed in ParaView from the length of neighbor edges and dihedral angles between normals of neighbor facets [11] while it is deduced from the 'Mean Curvature Normal Operator' in

Fig. 5a Extracted isocontour from a spherical function ($\frac{R}{\Delta x} = 4$).

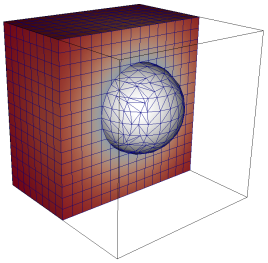

Fig. 5b Triangles generated with a marching cube method.

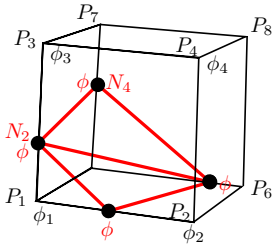

Fig. 5. Illustration of the marching cube method used to generate an isocontour representing the implicit interface contained in 3D data.

GTS [24]. The Gaussian curvature is computed with angle deficit methods both in ParaView and GTS [24] but with a more complex estimation of the area in the GTS library to deal with obtuse triangles. The implicit methods are based on diffuse approximation or Lagrange differentiation, as described in section 2.

Fig. 6a plots the maximum error on mean curvatures in log-scale as a function of the sphere resolution. Once again, relative errors on κ_K, κ_{min} and κ_{max} are very similar to mean curvature errors and are not shown for brevity. The explicit discrete methods of ParaView or GTS present both relative errors larger than 100% on curvatures. These discrete methods fail in estimating curvatures because the marching cube method generates a very irregular mesh of the sphere with non-uniform triangles (see Fig. 5a and section 3.3). Fitting methods com-

Fig. 6a Maximum errors

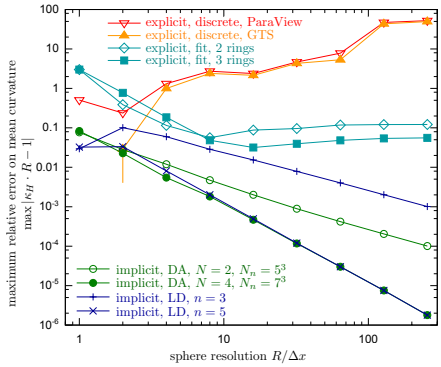

Fig. 6b Error distributions for $\frac{R}{\Delta x} = 8$

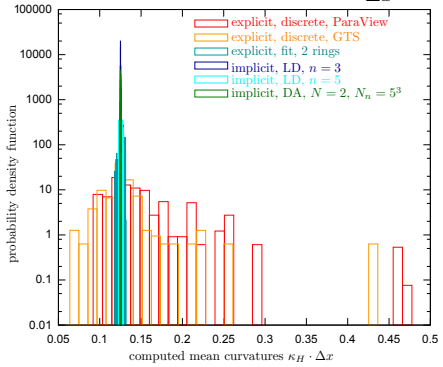

Fig. 6. Errors when computing mean curvatures of a spherical isocontour extracted from a 3D implicit function using a marching cube method.

pute reasonable estimations of curvatures with relative errors about 10% with sufficiently refined spheres ($R/\Delta x \geq 8$). Note that increasing the number of rings of neighbor nodes to fit paraboloids slightly increased the accuracy with such irregular meshes. Implicit methods significantly improves the accuracy to estimate curvatures of an isocontour generated with a marching cube method. The relative error on curvatures is below 1% for sufficiently refined spheres and about 10% at low resolution ($\frac{R}{\Delta x}$ or $\frac{\delta}{\Delta x} \simeq 1$).

The error distributions of the computed curvatures are also plotted for $R/\Delta x = 8$ in Fig. 6b. The large width and flatness of distributions for discrete methods demonstrate that these methods are inaccurate on a large number of nodes and not only at an isolated node. The implicit methods with a second-order diffuse approximation and Lagrange differentiation with higher order ($n \geq 5$) present the narrowest distributions and therefore highest accuracies.

3.3 Accuracy comparison using a regular triangle mesh

Compared to the previous test-case, the triangle mesh of the spherical isosurface is no more extracted by a marching cube method. It is '*artificially*' generated from the subdivision of an icosahedron constituted of $N_{\triangle,0} = 20$ equilateral triangles and 12 nodes. The i^{th} refined icosahedron has $N_{\triangle} = 4^i \cdot N_{\triangle,0}$ equilateral triangles. In this test, curvatures are then computed at the nodes of a very regular triangle mesh of a sphere. This regular mesh, constituted of $N_{\triangle}$ equilateral triangles of side Δx, is artificially positioned at a radius $\frac{R}{\Delta x} = (N_{\triangle}\sqrt{3}/16\pi)^{1/2}$ since $4\pi R^2 \simeq N_{\triangle} \cdot \mathcal{A}_{\triangle}$, with $\mathcal{A}_{\triangle} = \sqrt{3}\Delta x^2/4$ being the area of an equilateral triangle. The implicit spherical function is then defined with a stiffness $\frac{\Delta x}{\delta} = \frac{\Delta x}{R}$ to allow comparisons between implicit/explicit methods.

The maximum errors on mean and Gaussian are plotted in log-scale as a function of the sphere resolution $R/\Delta x$ in Fig. 7a. All explicit methods exhibit a better accuracy with such artificially generated regular meshes than with previous irregular meshes. If the built-in discrete method of ParaView reaches about 10% of accuracy, the GTS discrete method reaches accuracies under 1% and even under 0.01% for the mean curvatures. The accuracy is also increased for fitting methods with such regular meshes even if less accurate than the GTS discrete method. Implicit methods still reach highest accuracies on curvatures even if nodes of the *artificial* isosurface are located any-where within the stencil where ϕ is discretized, which requires more interpolations than previous test-cases. Second-order numerical schemes (DA with $N = 2$ and LD with $n = 3$) are 1^{st}-order accurate to compute curvatures whereas 4^{th}-order (LD with $n = 5$ or DA with $N = 4$) are 3^{rd}-order accurate. Implicit methods with high order schemes reach accuracies better than 1% for reasonable resolutions.

The error distributions at a moderate resolution $R/\Delta x \simeq 6.64$ are also shown in Fig. 7b. These distributions confirm that the errors are better bounded with regular meshes for explicit methods. The explicit discrete GTS methods and implicit methods are the most accurate since the error distribution is very narrow and centered around analytical solution. A shift from the analytical solution is

Fig. 7a Maximum errors Fig. 7b PDF for $\frac{R}{\Delta x} = \left(\frac{80\sqrt{3}}{\pi}\right)^{\frac{1}{2}} \simeq 6.64$

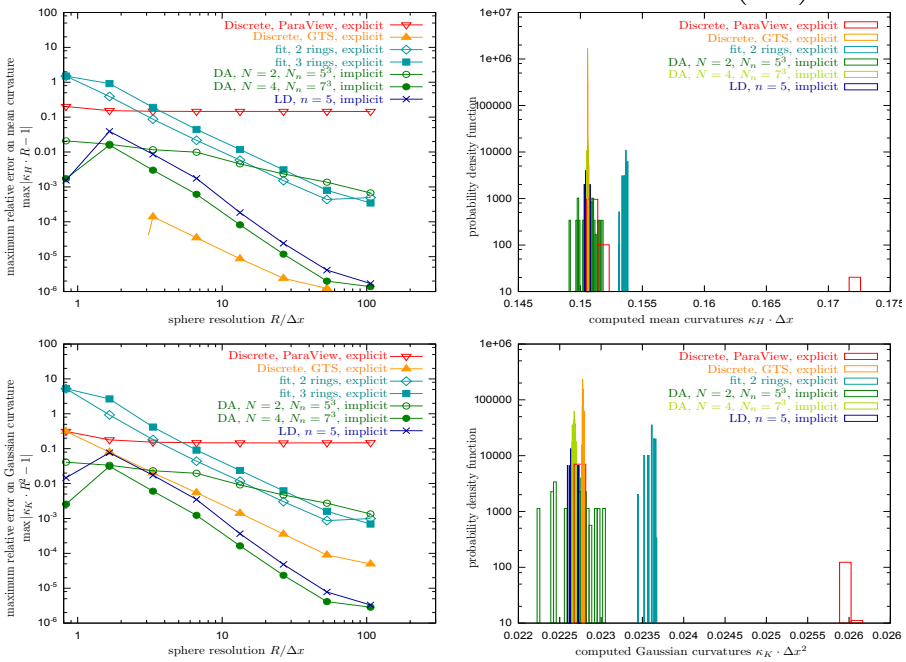

Fig. 7. Errors when computing mean and Gaussian curvatures of a regular triangle mesh (equilateral triangles artificially generated to avoid mesh irregularities of the isosurface).

present with the explicit fitting method and the computed curvatures are very dispersed with the discrete method implemented in ParaView.

The two previous test-cases show that explicit methods are really dependent on the regularity of the isosurface mesh. In opposite, the accuracy of implicit methods does not depend on the procedure used to extract the isocontour. It depends only on neighbor 3D scalar values and numerical differentiation schemes.

3.4 Accuracy assessment to measure principal directions

A donut is a nice geometry to test the proposed algorithm that computes principal directions. Indeed, this geometry presents elliptical points where the surface is convex ($\kappa_K > 0$), hyperbolic points where the surface is saddle shaped ($\kappa_K < 0$) and parabolic points ($\kappa_K = 0$) but does not contain umbillical points ($\kappa_{min} = \kappa_{max}$) where principal directions are undefined. An implicit function $\phi(x,y,z) = \tanh(\frac{\{((x^2+y^2)^{1/2}-R)^2+z^2\}^{1/2}-r}{\delta})$ is used to describe the torus with $r = \delta = \frac{R}{2}$. Fig. 8a illustrates the implicit toric function as a cheese chunk and principal directions computed on an extracted isocontour with a marching cube method for a resolution $\frac{r}{\Delta x} = 4$.

Fig. 8a Torus with $\frac{r}{\Delta x} = 4$

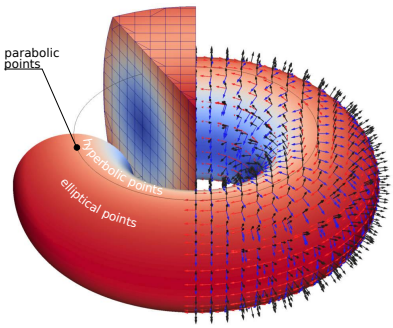

Fig. 8b Maximum angle error

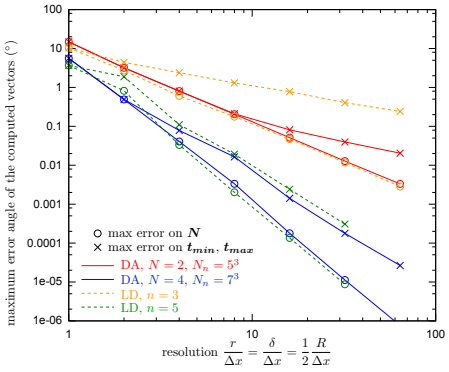

Fig. 8. Measurement of normals $\boldsymbol{n}$ and principal directions $\boldsymbol{t_{min}}$, $\boldsymbol{t_{max}}$ of a torus with implicit methods. The maximum angle error of all computed vectors are plotted in Fig. 8b for various numercial schemes.

An error between computed vectors and analytical solutions is then defined by $\|\boldsymbol{u} \wedge \boldsymbol{u}_{sol}\| = \sin(\widehat{\boldsymbol{u}, \boldsymbol{u}_{sol}})$ at each point of isocontours. Fig. 8b then plots the maximum angle error on computed normals and principal directions as a function of the torus resolution and for various differentiation schemes. The accuracy of implicit methods increases with the resolution and high order schemes. The largest angle errors are obtained with the 2^{nd}-order LD implicit method that uses the smallest stencil. They are below $1°$ with a second order diffuse approximation and below $0.1°$ with 4^{th}-order LD or DA schemes for sufficiently refined cases $\frac{\delta}{\Delta x} \geqslant 4$. The algorithm has been tested successfully through other cases that are not described here to be concise.

3.5 Speed assessment of the implicit method

The CPU time of implicit methods depends on the used numerical scheme (LD, DA and N) and the stencil size $N_n = n^3$. Lagrange differentiation methods are faster than diffuse approximation ones for the same stencil size (see Fig. 9a). LD methods allow to reach high orders with less CPU time.

The speed of the implicit method is also compared to those of discrete and fitting methods in Fig. 9b. All programs are written in $C\#$ for this speed comparison. The speed increases linearly with the number of points on the discrete sphere for all methods. Implicit methods with Lagrange differentiation reach the highest speeds. The fourth-order LD ($n = 5$) and 2^{nd}-order DA ($N_n = 3^3$) have comparable speeds to the built-in discrete method of the GTS library and the two-ring fitting method.

The curvature accuracy of these different methods is compared when computing Gaussian and mean curvatures of a same mesh extracted from a spherical implicit function with marching cubes like in section 3.2. Fig. 10a plots accuracy vs computation time for a coarse mesh and Fig. 10b for a finer mesh. These

Fig. 9a implicit methods

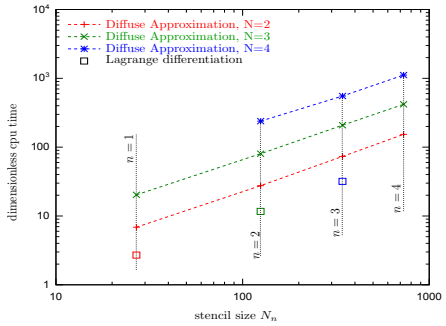

Fig. 9b implicit & explicit methods

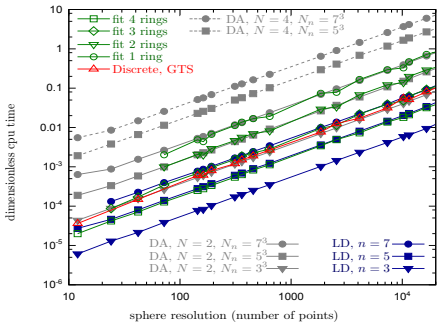

Fig. 9. Comparison of the computational speed to compute curvatures for different methods.

Fig. 10a coarse mesh (312 nodes, $\frac{R}{\Delta x} \simeq 4$) Fig. 10b fine mesh (4128 nodes, $\frac{R}{\Delta x} \simeq 15$)

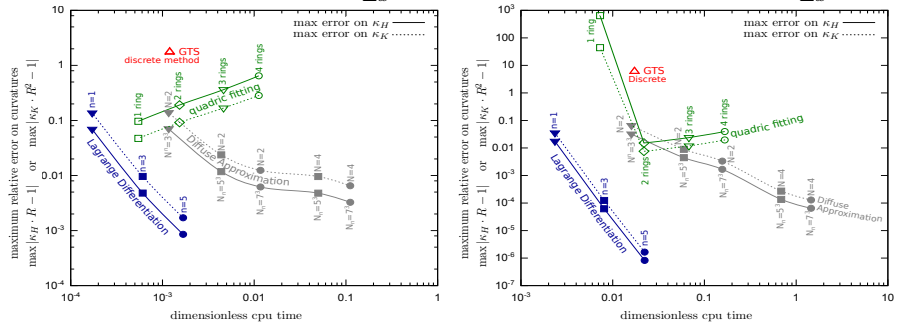

Fig. 10. Comparison diagram of computation time vs accuracy for various implicit and explicit methods. Curvatures are measured on a coarse and refined mesh generated from a spherical implicit function with marching cubes.

diagrams show that LD implicit methods reach the best compromises between speed and accuracy. Discrete method and the 1-ring fitting method fail in predicting accurately curvatures at some nodes. DA implicit methods allow to reach higher accuracies than quadric fitting method but may require an additional cpu cost.

4 Applications

These methods are applied to determine curvatures of expanding and imploding flames. The flame images are obtained from Direct Numerical Simulations using the H-allegro in-house software [1] on a supercomputing infrastructure (PRACE).

4.1 Laminar flames

An unburnt methane-air mixture at $T_u = 480K$ is ignited at atmospheric pressure with a Gaussian profile in temperature and composition. The adiabatic flame temperature of such a mixture is $T_b = 2260K$ and the planar thermal flame thickness is $\delta_L^0 = 202\mu m$. The computational domain is a cube of $1.5cm$ width discretized with a Cartesian grid of $(336)^3$ points, which implies a sufficient refinement to solve the flame propagation ($\delta_L^0/\Delta x \simeq 4.5$).

Curvatures of the flame front are measured in the simulation results along the flame propagation with different methods. To determine flame curvatures, isocontours of a $569K$ temperature (progress variable of 5%) are first extracted with a marching cube method [19]. The ParaView software [31] is used to handle the multiple-files of the parallel solution (512 files for one time-step) and the complex topology of isocontours with a triangle mesh. Fifteen quasi-spherical isocontours are extracted along the flame propagation; the flame evolves from a small radius $R_S \simeq 12\delta_L^0$ to a larger radius $R_S \simeq 150\delta_L^0$.

Figure 11 shows the computed local flame radius $R_H = 1/\kappa_H$ based on mean curvatures as a function of the area based flame radius $R_S = \sqrt{S/(4\pi)}$. Curves for the Gaussian curvatures are not shown because too similar. At each flame radius, the discrete methods of ParaView or GTS are inaccurate, because of the irregularity of the mesh. The scattering of the local radius is so large that we did not plot it out for comparisons. With fitting methods, the scattering is reduced in particular when several rings to fit parabolas are used. Nevertheless, the dispersion is still very high even with a 4 ring stencil. Implicit methods with large stencils ($N_n \geqslant 5^3$ points) points significantly improves the accuracy when predicting the local flame radius based on the mean or Gaussian curvatures, which makes curvature analysis in flows possible. The small remaining scattering at large radii is attributed to the effect of boundary conditions and not to the inaccuracy of the proposed method.

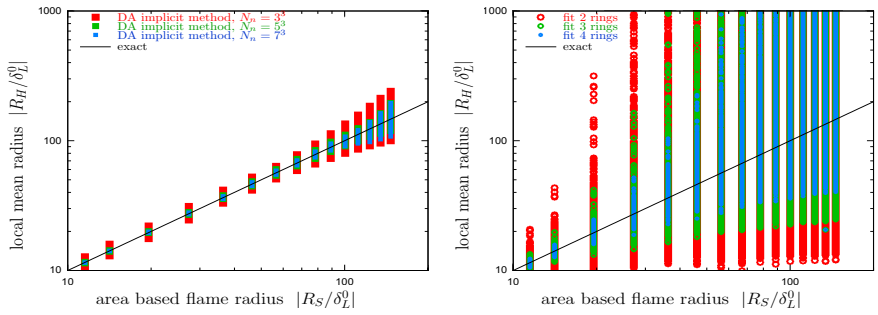

Fig. 11. Comparison of the measured local radii to the global radii based on the whole surface when the flame propagates for DA implicit and fitting methods.

Probability density functions (PDF) of Gaussian and mean curvatures are plotted in Fig. 12 at the middle time of the simulation ($t = 0.785ms$). At this stage, the flame is quasi-spherical with a radius $R \simeq -0.3cm$. The probability density function and cumulative function are respectively expected to be close to a Dirac function and a Heaviside function. For the implicit method, the probability density is as expected a peak centered around $\kappa_H = 1/R$ or $\kappa_K = 1/R^2$ and the cumulative probability is the stiffest step compared to the fitting and discrete methods. The absence of peaks on the PDF for fitting and discrete methods shows that explicit methods are really inaccurate to estimate the curvatures of this stiff flame ($\delta_L^0/\Delta x \simeq 4.5$).

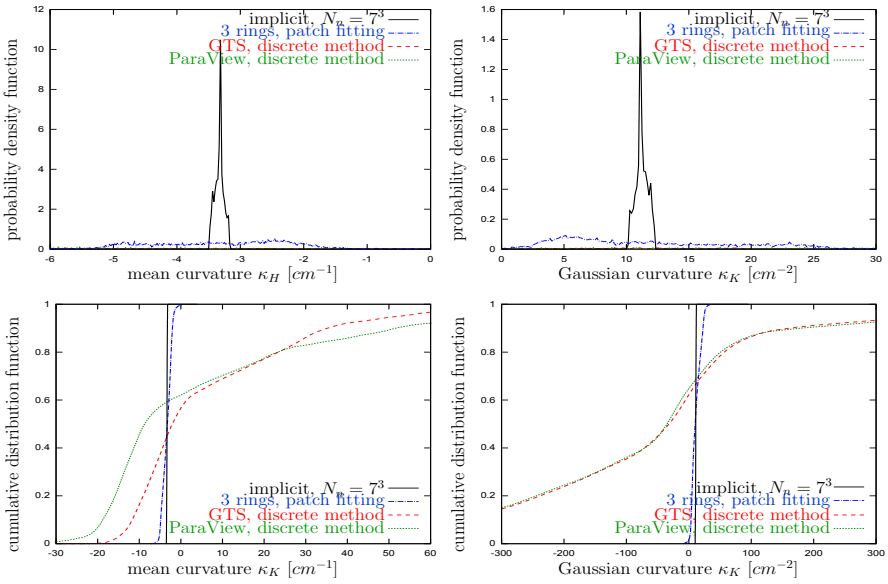

Fig. 12. Probability density function and cumulative distribution function of measured Gaussian, mean, minimal and maximal curvatures of a quasi-spherical flame ($t = 0.785ms$). Note that only the implicit method has a PDF close to a Dirac function and a cumulated probability close to a Heaviside function.

4.2 Turbulent flames

A turbulent imploding hydrogen flame premixed with air diluted with 20% of steam is ignited by a Gaussian profile of temperature. After a transient, the flame propagates inward and becomes highly wrinkled. An isocontour of temperature (corresponding to a progress variable value of 50%) is extracted with a marching cube method at $1.5ms$ after the ignition. The mean, Gaussian, minimum and maximum curvatures of this isocontour are then computed at every nodes of the

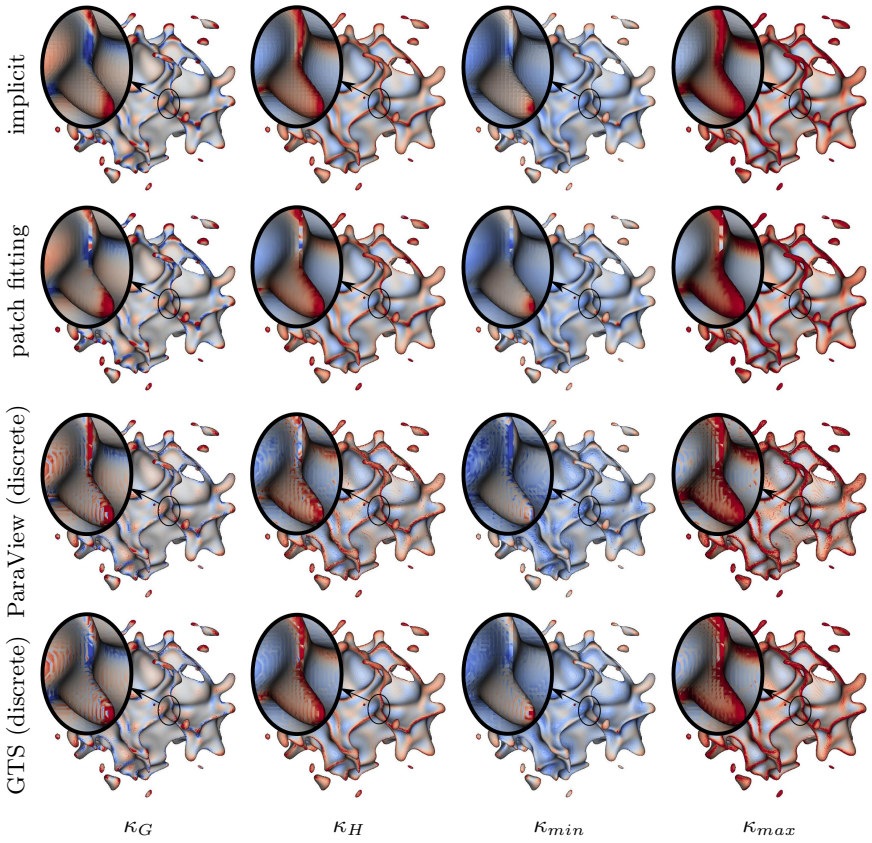

Fig. 13. Test of different methods to compute Gaussian, mean, minimal and maximal curvatures on a triangular unstructured mesh. Curvatures are plotted in colors from blue to red ($\kappa_H, \kappa_{min}, \kappa_{max} \in [-50; 50 cm^{-1}]$).

mesh making use of i) a 2^{nd}-order DA implicit method with $N_n = 7^3$, ii) a fitting method with a 3 ring patch, iii) the explicit discrete method of GTS and iv) the explicit discrete method of ParaView. Results are compared in figure 13.

If all methods seem to compute very similar curvatures, a careful look shows numerous oscillations for the discrete methods (compare zooms on κ_H in Fig. 13). Both the fitting method (with a large stencil) and the DA implicit method show very similar results. However, the fitting method fails at some node locations to fit a parabola; this method also generates visible oscillations near highly curved surface compared to the implicit method (cf. zooms). However, the similitude between results confirms the ability of the implicit method to compute mean, Gaussian, minimum and maximum curvatures, even for complex geometries. From previous test-cases, it may be expected that the implicit method give the most accurate estimates of curvatures. Explicit methods show a more acceptable prediction of curvatures compared to the inaccuracies observed on the laminar

flame, this is attributed to the quality of the isosurface mesh that is better defined in the turbulent case with a different progress variable (50% vs 5%) and a more refined flame ($\delta_L^0/\Delta x \simeq 7.8 > 4.5$).

Figure 14 shows some examples of local analyses that may be conducted to study turbulent flames with implicit methods. Some computed principal directions of the previous hydrogen imploding flame are shown in Fig. 14a. The implicit method may be used to measure other flame properties from normals and curvatures. It may for instance be used to measure local consumption of the gaz mixture, local flame speeds, local flame stretches. Fig. 14b shows a scatter plot of the local consumption speed as a fonction of mean and principal curvatures for a turbulent methane-air expanding flame, which highlights a link between gaz consumption and mean flame curvatures in this case.

Fig. 14a principal directions

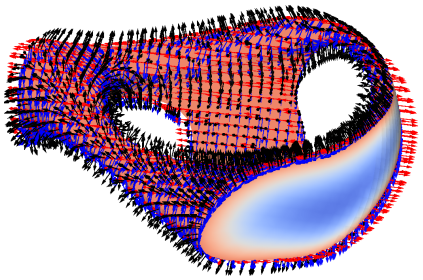

Fig. 14b local mixture consumption

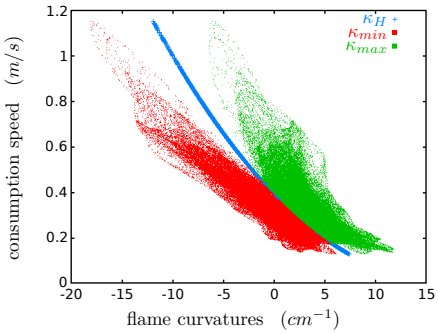

Fig. 14. Example of local flame analysis using implicit methods.

5 Conclusion

A computational method has been presented to compute curvatures and principal directions of implicit surfaces from 3D scalar data. Curvatures are computed at any vertex whithin the stencil where the implicit surface is discretized. This implicit method makes use of high-order Diffuse Approximation (DA) or Lagrange differentation (LD) schemes to interpolate/differentiate the 3D scalar function. Implicit methods are compared to standard explicit methods (surface fitting, discrete methods) when computing curvatures along isocontours representing the implicit surface.

It is evidenced through numerous test cases that explicit methods fail in measuring accurately curvatures of surfaces with irregular meshes. However, the extraction of surfaces with complex topologies is not trivial and commonly used techniques like marching-cube methods do not always generate regular meshes. It is shown that implicit methods do not require regular meshes and may reach

high accuracy, which makes these methods interesting to analyse curvatures of isosurfaces from 3D data. If the stiffness of the implicit function is known and not too high, the maximum error on curvatures with implicit methods may be estimated from Fig. 6a. The implicit method is also found to be competitive with discrete methods in terms of computational speed. A 2^{nd}-order DA or LD schemes of higher order are recommended when computing curvatures to reduce the error without requiring much cpu overhead.

The proposed implicit method has been successfully applied to measure properties of flame surfaces based on normals and curvatures and shows very promising results to conduct accurate local 3D analysis of wrinkled flames (local consumption speed, displacement speeds, curvatures and stretches of the flame). A perspective of this work would be to extend this methodology to implicit surfaces defined by stiffer implicit functions using non-oscillatory schemes for interpolation/differentiation since discontinuous 3D scalar data may be encountered for instance in multiphase flows or medical images.

A Derivation of formulas for principal directions of implicit surfaces

The methodology of Lehmann *et* al. [18] is used to build the curvature tensor in the $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{n})$ frame. In this frame, the orthogonal projector T onto the tangent space and the Hessian matrix of ϕ are respectively:

$$T = Id_3 - \boldsymbol{n}\cdot\boldsymbol{n}^t = \begin{pmatrix} 1&0&0\\ 0&1&0\\ 0&0&0 \end{pmatrix} \quad and \quad \nabla^2\phi = \begin{pmatrix} \phi_{uu} & \phi_{uv} & \phi_{uN}\\ \phi_{vu} & \phi_{vv} & \phi_{vN}\\ \phi_{Nu} & \phi_{Nv} & \phi_{NN} \end{pmatrix}_{(\boldsymbol{u},\boldsymbol{v},\boldsymbol{n})}$$

The curvature tensor has then a simple expression in the normal frame:

$$E = \frac{T\cdot\nabla^2\phi\cdot T}{|\phi_n|} = \begin{pmatrix} \frac{\phi_{uu}}{|\phi_n|} & \frac{\phi_{uv}}{|\phi_n|} & 0\\ \frac{\phi_{uv}}{|\phi_n|} & \frac{\phi_{vv}}{|\phi_n|} & 0\\ 0 & 0 & 0 \end{pmatrix}_{(\boldsymbol{u},\boldsymbol{v},\boldsymbol{n})} \tag{12}$$

Its expression is far more complicated in an arbitrary basis $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})$ where we could not find eigenvectors. In the normal frame, main eigenvalues and corresponding eigenvectors of the curvature tensor are then found equal to:

$$\kappa_1 = K_h - \sqrt{|\kappa_h^2 - \kappa_k|}\cdot\zeta \qquad \kappa_2 = K_h + \sqrt{|\kappa_h^2 - \kappa_k|}\cdot\zeta \tag{13}$$

$$\boldsymbol{t_1} = \frac{1}{D_1}\begin{bmatrix} \phi_{uv}\\ \kappa_1\phi_n - \phi_{uu}\\ 0 \end{bmatrix}_{\boldsymbol{u},\boldsymbol{v},\boldsymbol{n}} \qquad \boldsymbol{t_2} = \frac{1}{D_2}\begin{bmatrix} \kappa_2\phi_n - \phi_{vv}\\ \phi_{uv}\\ 0 \end{bmatrix}_{\boldsymbol{u},\boldsymbol{v},\boldsymbol{n}} \tag{14}$$

with $\kappa_k = \frac{\phi_{uu}\phi_{vv}-\phi_{uv}^2}{\phi_n^2}$, $\kappa_h = \frac{\phi_{uu}+\phi_{vv}}{2|\phi_n|}$, $D_1 = D_2 = \sqrt{\phi_{uv}^2 + (\kappa_1\phi_n - \phi_{uu})^2}$ and $\zeta = \pm 1$.

Since derivatives in the ($\boldsymbol{u}$,$\boldsymbol{v}$,$\boldsymbol{n}$) basis are linked to the derivatives in the default basis ($\boldsymbol{x}$,$\boldsymbol{y}$,$\boldsymbol{z}$) and coordinates of $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{n}$:

$$\phi_n = \boldsymbol{\nabla}\phi \cdot \boldsymbol{n} = \phi_x n_x + \phi_y n_y + \phi_z n_z = \|\boldsymbol{\nabla}\phi\| = |\phi_n| \tag{15a}$$

$$\begin{aligned}\phi_{uv} = \boldsymbol{u}^t \cdot \nabla^2\phi \cdot \boldsymbol{v} &= \phi_{xx}u_x v_x + \phi_{yy}u_y v_y + \phi_{zz}u_z v_z \\ &+ \phi_{xy}(u_x v_y + u_y v_x) + \phi_{xz}(u_x v_z + u_z v_x) + \phi_{yz}(u_y v_z + u_z v_y)\end{aligned} \tag{15b}$$

$$\phi_{uu} = \boldsymbol{u}^t \cdot \nabla^2\phi \cdot \boldsymbol{u} \tag{15c}$$

$$\phi_{vv} = \boldsymbol{v}^t \cdot \nabla^2\phi \cdot \boldsymbol{v} \tag{15d}$$

Replacing ϕ_n, ϕ_{uv}, ϕ_{uv} and ϕ_{uv} into curvature expressions (2b), (3b) and making some formal simplication, we recover the intrinsic expressions of curvatures (2a), (3a) that are independent on the choice of $\boldsymbol{u}$,$\boldsymbol{v}$. To sum up, principal curvatures do not depend on the choice of ($\boldsymbol{u}$,$\boldsymbol{v}$) whereas principal directions depend on their coordinates. A bad choice of ($\boldsymbol{u}$,$\boldsymbol{v}$) may then conduct to null vectors and then bad estimations. To circumvent bad choices, we then propose a criteria on ζ to ensure that $\boldsymbol{t_1}$ and $\boldsymbol{t_2}$ are non-null vectors in section 1.

To better understand the role of the parameter ζ in avoiding degeneracies, it is advised to compute curvatures and principal directions by hand with $\zeta = \pm 1$ at $u = v = 0$ and $n = |R|$ for the following case studies:

- $\phi(u, v, n) = u^2 + n^2 - R^2 \Rightarrow$ cylinder of axis $\boldsymbol{v}$ that requires $\zeta = +1$
- $\phi(u, v, n) = R^2 - u^2 - n^2 \Rightarrow$ cylinder of axis $\boldsymbol{v}$ that requires $\zeta = -1$
- $\phi(u, v, n) = v^2 + n^2 - R^2 \Rightarrow$ cylinder of axis $\boldsymbol{u}$ that requires $\zeta = -1$
- $\phi(u, v, n) = R^2 - v^2 - n^2 \Rightarrow$ cylinder of axis $\boldsymbol{u}$ that requires $\zeta = +1$
- $\phi(u, v, n) = \pm u * v + n \Rightarrow$ saddle shaped surface ($\zeta = \pm 1$)

This work was granted access to the HPC resources of the RZG of the Max Planck Society made available within the Distributed European Computing Initiative by the PRACE-2IP, receiving funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° RI-283493. The research leading to these results has received funding from the European Research Council under the ERC grant agreement n° 247322, GREENEST. The authors thank Xiang He for his help in code development and Bruno Denet for his feedback in using the GTS library.

References

1. Albin E, D'Angelo Y. 2012. *Assessment of the Evolution Equation Modelling approach for three-dimensional expanding wrinkled premixed flames.* Combustion and Flame 159(5) pp. 1932–1948.
2. Albin E, D'Angelo Y, Vervisch L. 2010. *Using staggered grids with characteristic boundary conditions when solving compressible reactive Navier-Stokes equations.* International Journal for Numerical Methods in Fluids 68(5) pp. 546–563.
3. Albin E, D'Angelo Y, Vervisch L. 2011. *Flow streamline based Navier–Stokes characteristic boundary conditions: modeling for transverse and corner outflows.* Computers & Fluids 51 (1) pp. 115–126.

4. Brackbill JU, Kothe DB, Zemach C. 1992. *A continuum method for modeling surface tension*, Journal of computational physics 100(2) pp. 335–354.
5. Cazals F and Faugère JC, Pouget M, Rouillier F. 2006. *The implicit structure of ridges of a smooth parametric surface*, Computer Aided Geometric Design 23 (7). pp. 582–598.
6. Chang YC, Hou TY, Merriman B, Osher S. 1996. *Level Set Formulation of Eulerian Interface Capturing Methods for Incompressible Fluid Flows.* Journal of Computational Physics 124(2) pp. 449-464.
7. Chauveau C, Birouk M and Gökalp I. 2011. *An analysis of the d^2-law departure during droplet evaporation in microgravity.* International Journal of Multiphase Flow 37(3) pp. 252–259.
8. D'Angelo Y, Joulin G, Boury G. 2000. *On model evolution equations for the whole surface of three-dimensional expanding wrinkled premixed flames.* Combustion Theory and Modelling 4(3) pp. 317–338.
9. Denet B. 2004. *Nonlinear model equation for three-dimensional Bunsen flames.* Physics of Fluids 16(4) pp. 1149–1155.
10. Douros I, Buxton BF. 2002. *Three-dimensional surface curvature estimation using quadric surface patches.* Scanning.
11. Dyn N, Hormann K, Kim SJ and Levin D. 2001. *Optimizing 3D triangulations using discrete curvature analysis.* Mathematical methods for curves and surfaces pp. 135–146.
12. Gatzke T, Grimm CM. 2006. *Estimating curvature on triangular meshes.* International Journal of Shape Modeling 12(1) pp. 1–28.
13. Giuliani D. 2005. *Gaussian curvature: A growth parameter for biological structures.* Mathematical and computer modelling 42(11) pp. 1375–1384.
14. Goldman R. 2005. *Curvature formulas for implicit curves and surfaces.* Computer Aided Geometric Design 22(7) pp. 632–658.
15. Hameiri E, Shimshoni I. 2003. *Estimating the principal curvatures and the Darboux frame from real 3-D range data.* Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 33(4) pp. 626–637.
16. Jeffreys H, Jeffreys BS. 1988. *Methods of Mathematical Physics*, §9.011 Lagrange's Interpolation Formula, Cambridge University Press, Cambridge, England, pp. 261.
17. Lebas R, Menard T, Beau PA, Berlemont A, Demoulin FX. 2009. *Numerical simulation of primary break-up and atomization: DNS and modelling study.* International Journal of Multiphase Flow 35(3) pp. 247–260.
18. Lehmann N, Reif U. 2012. *Notes on the curvature tensor.* Graphical Models 74. pp 321–325.
19. Lorensen WE, Cline HE. 1987. *Marching cubes: A high resolution 3D surface construction algorithm.* ACM Siggraph Computer Graphics 21(4) pp. 163–169.
20. Macklin P, Lowengrub J. 2006. *An improved geometry-aware curvature discretization for level set methods: application to tumor growth.* Journal of Computational Physics 215(2) pp. 392–401.
21. Marchandise E, Geuzaine P, Chevaugeon N, Remacle JF. 2007. *A stabilized finite element method using a discontinuous level set approach for the computation of bubble dynamics.* Journal of Computational Physics 225(1) pp. 949-974.
22. McIvor AM, Valkenburg RJ. 1997. *A comparison of local surface geometry estimation methods.* Machine Vision and Applications 10(1) pp. 17–26.
23. Ménard T, Tanguy S, Berlemont A. 2007. *Coupling level set/VOF/ghost fluid methods: Validation and application to 3D simulation of the primary break-up of a liquid jet.* International Journal of Multiphase Flow 33(5) pp. 510–524.

24. Meyer M, Desbrun M, Schröder P, Barr AH. 2003. *Discrete differential-geometry operators for triangulated 2-manifolds*. Visualization and mathematics III pp. 35–57.
25. Musuvathy S, Cohen E, Damon J and Seong JK. 2011. *Principal curvature ridges and geometrically salient regions of parametric B-spline surfaces*. Computer-Aided Design 43 (7). pp 756–770.
26. Osher S, Fedkiw R. 2003. *Level set methods and dynamic implicit surfaces*. Springer.
27. Page DL, Sun Y, Koschan AF, Paik J, Abidi MA. 2002. *Normal vector voting: crease detection and curvature estimation on large, noisy meshes*. Graphical Models 64(3) pp. 199–229.
28. Peng J, Li Q, Kuo CCJ, Zhou M. 2003. *Estimating Gaussian curvatures from 3D meshes*, Electronic Imaging pp. 270–280.
29. Popinet S. 2004. *The GNU triangulated surface library*.
30. Prax C, Sadat H, Dabboura E. 2007. *Evaluation of high order versions of the diffuse approximate meshless method*. Applied mathematics and computation 186(2) pp. 1040–1053.
31. Squillacote AH. 2007. *The ParaView guide: a parallel visualization application*. Kitware.
32. Sussman M, Puckett EG. 2000. *A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows*. Journal of Computational Physics 162(2) pp. 301–337.
33. Thirion JP, Gourdon A. 1992. *The 3D marching lines algorithm and its application to crest lines extraction*. Research report.
34. Treece GM, Prager RW, Gee AH. 1999. *Regularised marching tetrahedra: improved iso-surface extraction*, Computers & Graphics 23(4) pp. 583–598.
35. Trott M. 2004. *The mathematica guidebook for graphics*. Springer.
36. Yan B, Li B, Baudoin E, Liu C, Sun ZW, Li ZS, Bai XS, Aldén M, Chen G, Mansour MS. 2010. *Structures and stabilization of low calorific value gas turbulent partially premixed flames in a conical burner*. Experimental Thermal and Fluid Science 34(3) pp. 412–419.
37. Yoo CS, Im HG. 2007. *Characteristic boundary conditions for simulations of compressible reacting flows with multi-dimensional, viscous and reaction effects*. Combustion Theory and Modelling 11(2) pp. 259–286.
38. Yu R, Bai XS. 2013. *Direct numerical simulation of lean hydrogen/air auto-ignition in a constant volume enclosure*, Combustion and Flame 160 (9) pp. 1706–1716.